Issue 33 | May 2011

blender art MAGAZINE

Blender learning made easy...

Everything but the Kitchen Sink

www.blenderart.org

Antea -by Martin Eschoyez

CONTENTS

EDITOR - Gaurav Nawani
MANAGER/EDITOR - Sandra Gilbert
WEBSITE - Nam Pham
DESIGN - Gaurav Nawani

PROOFERS
Brian C. Treacy
Bruce Westfall
Daniel Hand
Daniel Mate
Henriël Veldtmann
Joshua Leung
Joshua Scotton
Kevin Braun
Mark Warren
Noah Summers
Patrick ODonnell
Phillip
Ronan Posnic
Scott Hill
Wade Bick
Valérie Hambert

WRITERS
Thomas Baron
Pedro BastosTeemu Vilén
Alexander Weide
Stephan Rodriguez
Darcy Brooker
Brian DeVincentis
Enrique Sahagun
Nathan Moyer
Sandra Gilbert
Gaurav Nawani

COVER ART
Antea -by Martin Eschoyez

Upcomming Issue 34 - "Rigs, Add-ons and more..."

* Characters, individual body parts (i.e. feet, hands, etc).
* Animals and reptiles.
* Vehicles and other mechanical rigging, any other use for rigging.
* Add-ons and the python behind them.
* How to extend blender through python.
* New builds/patches/branches.

EDITORIAL

Sandra Gilbert

Manager/Editor

"When I announced the theme for this issue "Everything but the kitchen sink", I was jokingly threatened with massive numbers of sink projects."

Over the years we have covered a number of fun and interesting topics. But I must admit, my favorite issues are always the ones where we focus on the artists and their projects.

I love hearing what everyone is up to and seeing how they go about it. There is always something new to be learned from watching others work. Then of course there is the added inspiration of seeing something done that you hadn't considered before.

That of course does lead to one minor problem, at least for me. It creates far more creative ideas than I have time for... LOL, what a lovely problem to have.

Anyhow, I am tickled to report that this issue is filled with inspiring projects from talented users on a wide range of topics. Although I must admit to a tiny bit of disappointment.

When I announced the theme for this issue "Everything but the kitchen sink", I was jokingly threatened with massive numbers of sink projects.

And I didn't get even one. I do believe that some of you owe me a sink. :P

But even though there are no promised sink projects, we have plenty to keep you busy for quite a while.

So let's get to it!

Izzy Speaks: The Journeyman Project Tribute

Izzy Speaks

I love hearing about new blender projects. Everyone has their own pet ideas on what would be cool to do or accomplish using blender. Some projects are by nature short and sweet and produce cool results or new techniques that can be shared with others. Then there are the labors of love. Projects that are destined to take months of dedicated work.

The Journeyman Project Tribute is one such example. Not only is the Journeyman Project Tribute a labor of love, it is a rather ambitious one at that. Andrew Curry is leading an enthusiastic team who have decided to remake the classic game "The Journeyman Project" by Presto Studios. The project is currently split into three efforts:

* A prequel novel

* A port to Windows of "Pegasus Prime", a remake of Journeyman by the original studio.

* A retelling of Journeyman using Blender to model and render the video footage, "Journeyman HD".

At the moment, there are several members hard at work on the novel and port teams, but so far the vast majority of the HD version has been done by Andrew. Another team member is busy modeling and rigging three characters from the game. Andrew has been keeping a blog of the modeling work he has finished so far. I wandered over to take a look, and let me tell you, it looks amazing. The details and texture work is beautiful. In addition to still images, he also has a growing number of test videos to show off his progress.

Andrew has told me that one of his goals is to keep the project open source and completely create the HD footage in Blender. If his current tests are any indication, the finished result should be quite impressive.

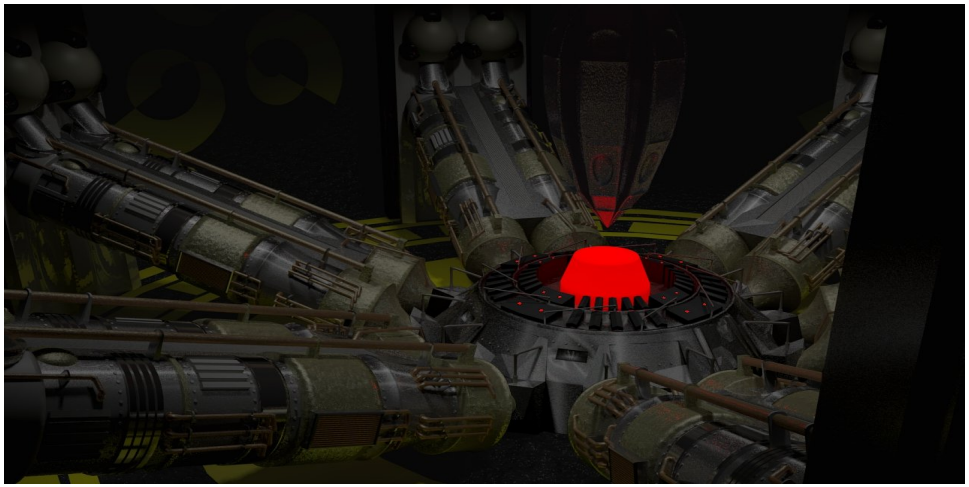
In an effort to keep the tribute as close as possible to the classic original as well as head off any legality issues, Andrew has been in touch with several members of Presto Studios, including the president of the studio. Presto Studios have given Andrew and his team their blessings for their open source (non-commercial) tribute. Even more fun, the original modeler is keeping an eye on Andrew's blog. No pressure there :P

The Journeyman Project Tribute looks to be well organized and planned out, making it a fun project to keep an eye on as they progress their way through all their goals.

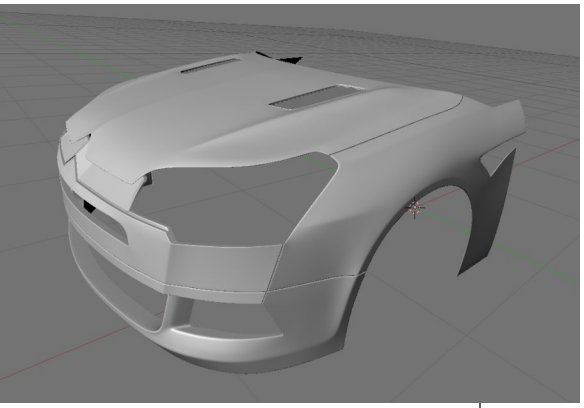
Andrew is looking for additional volunteers to help bring this project to completion. You can contact him for further information •

andrewcurry2@googlemail.com

http://andyjourneyman.blogspot.com/



3D WORKSHOP : Modeling a High-end Car



by - **Thomas Baron**

Introduction

No matter whether you are an amateur or a pro, this tutorial aims to teach the work flow and every move you need to know to achieve a good car. It's based on Blender, to allow everyone do it without requiring access to high-priced or pirated software. But it can apply to virtually any 3D package.

I based this work on my latest project, a coupe-convertible Citroën C5. If you're allergic to this brand, or to cars in general, don't worry as this material applies to any manufactured product regardless of whether it's big or small, cheap or high-end.

This tutorial is an on-going series of posts. It starts from the very beginning of modeling and covers every aspect: hard surface modeling with a focus on high-end quality and accuracy, best meshing practices to create a nice polyflow and tips to solve the most common issues. In addition to the body, this tutorial will deal with the creation of other parts such as tires, rims, wheel details, windows, grilles, chrome and rubber trim as well as a fully detailed interior.

This project is a common example of an amateur project. The idea came from the 2009 Real World Racer contest held by Scratch Made Cars (SMCars). The rule were simple: "do your dream racing machine". I choose to convert my daily ride, a Citroën C5 Tourer, to a Deutschen Tourenwagen Masters (DTM) coupé. During this project I wanted to do a "civil" version as well to put in the background of the render but a lack of time prevented me from doing so. This year, I entered another SMC contest called Topless, about ... topless cars. For this contest I did a Hennessey Venom GT, which I posted here, but I've had another idea. A convertible, civil version of my C5. I just wanted to make things a bit more difficult, by doing a CC version (coupé – convertible). So here's my CC5. Yes I know, the name isn't great.

Concept

The first step is creating a concept drawing of my idea. I always do it in 3D, without paper. So I create what is usually called a "spline cage", which means I have to draw the main curves of the car using Blender. However I don't use splines for that, I just use a regular mesh without any faces. I based the drawing on the C5 Tourer (station wagon) blueprints. You can find a previous tutorial of mine on how to setup blueprints here: Blueprints setup with Blender.

The major steps are :

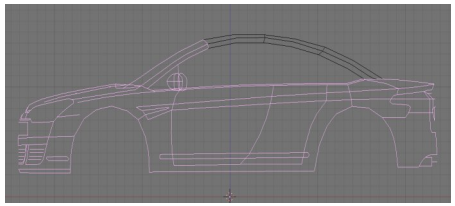
1. Find blueprints on the Internet. www.SMCars.net is a very good entry point, with thousands of blueprints available

2. In case you don't find blueprints, try to gather some side, front and rear views. Top photographs are quite hard to find, but they're not mandatory.

3. Collect as many reference pictures as you can find. Try to avoid CG pictures unless they're official, because you don't want to reproduce the mistakes from other CG artists.

4. There are many good reference picture sites listed at www.SMCars.net but try to grab high resolution photos and be careful about picking the the right model year, the right version, etc Setup blueprints (or pictures) like I explained in my tutorial

5. Draw a cage by creating a mesh without any faces. Basically, you create a plane, delete 3 vertices out of 4, place the last one at a particular spot, and extrude it several times to create an edge. Here's the resulting drawing.



The features I draw are:

1. The separation lines between the various panels.

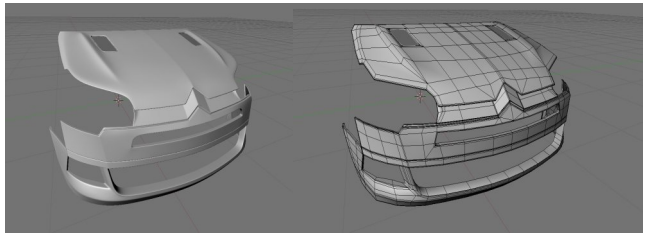
2. The major shapes of the car: wheel arches, folds, bumps, openings etc

3. Some extra curves which are useful to better define the shapes e.g. to show the roundness of a fender.

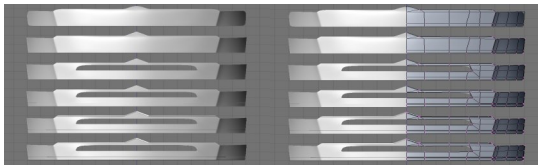
The main point as this stage is keeping things as clean and accurate as possible. So I rotate a lot around my cage in perspective mode to compare it with reference pictures. Having the actual car in front of my house helps a lot, so you may want to consider modeling your own car (or your father's etc...) as one of your first projects. If it wasn't my car, browsing the Internet is an acceptable solution. Choose high resolution pictures, but make sure you get the very same version that you are modelling.

An introduction to modeling: front bumper

My spline cage needs more refining but I chose instead to begin actual modeling. This part won't go into a lot of details as the goal is to give you an overview of the workflow. The next chapter will be much more detailed.



The trick is to model the major surfaces first, then add details. For example, for the two pieces between the bonnet and front bumper, I do several steps.



From top to bottom:

1. I start from my cage to make a simple but complete surface. The only detail at this stage is the almost vertical edge that I have already sharpened. At this stage you must be careful to have very smooth surfaces because it will be difficult to change later. The mesh topology is important as well, try to keep it simple and well balanced.

2. I add rows of vertices along the edges to sharpen the edges of the body panel where it will be given thickness later. You should have a regular mesh topology based on squares from the previous step, so this step will be simple. I take care about keeping constant spacing (0.005 BU or 5mm for my scale) between the edges and the additional rows in order to have a regular curvature of the edges later.

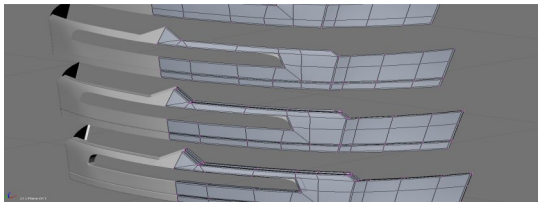
3. I create the hole in the upper air intake, relying on the mesh topology in place. Care must be taken to have a row of faces along the edge of the hole, just to have a smooth edge.

4. I cut the bottom (hard to see on the picture, sorry) benefiting from having reserved a line of vertices along the separation in my first step. That way, I simply add two rows of vertices on either side of that line, at 1.5mm each. By eliminating the initial line, I find myself with a regular spacing of 3mm, and the sharpened almost-vertical edge is continuous from one panel to another.

5. I add the thickness for the sheet metal panels by extruding edges. I extrude in 2 times, first for the 3mm radius of curvature of the edge of the panel sheet. The second extrusion is intended to have spaces between body panels rendered dark (7mm to a give a final of 1cm)

6. I finish the details of the body on top of this piece (it will be behind the lower chrome trim of the brand logo) and add the thickness to the bottom.

Here's a perspective shot to better see details of the last steps.

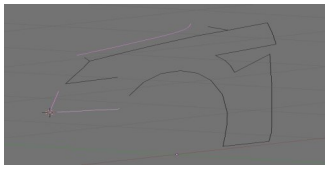


Detailed modeling: the front fender

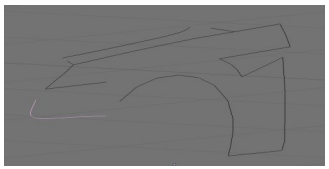
The next step is modeling the front fender. For this step, I will try to go as detailed as I can in my mesh construction workflow.

First, I the gather the edges that this fender will have in common with its neighbors (the hood and front bumper). In edit mode for each piece, I duplicate the whole edge and separate it. I also duplicate the relevant edges from my cage. I end up with several parts as shown below. The pink parts are from the hood and bumper, note that they are already smoothed (because there is level 3 SubSurf modifier active) while the black part is from

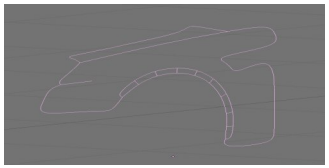
the cage and isn't smoothed.



I now have to merge all the edges together. I start by first merging the corner of the two edges from the bumper. Please note that to do so, the horizontal edge has been raised by 3mm to align with the other edge from the blueprint, while the other edge hasn't moved yet. So there is already a gap between the front bumper and the fender below the horizontal edge, while there is no gap for the other edge. We'll take care of it later, when the surface will be completed.

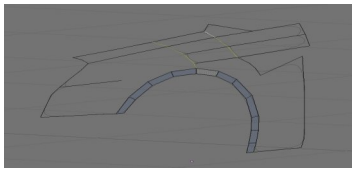


During the next step, I merged all the edges, which included merging the corner vertices. I removed the extra vertices at 5mm from each corner as they were responsible for the corners being sharp on the previous picture. With them removed, the corners are smoothed. It will be easier to build the surfaces with fewer vertices, and it won't be difficult to put them back later by adding extra rows as I described for the bumper modeling.

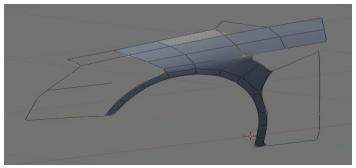


We now have a single mesh, made solely from edges. It's now time to prepare some surfaces.

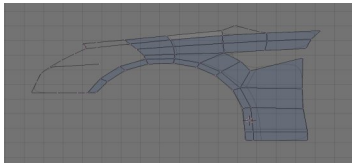
What I did in the below picture is extrude the wheel arch to create the flat side around it. As stated for the bumper modeling, I don't care yet about sharpening any edge. I also extruded some longitudinal edges and created the transverse, relevant edges (selected in the below pic, so they appear yellow) I always check that all the edges are smooth at every time.



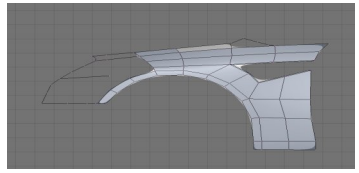
Now we can create the faces. Remember to always create rows of faces along all edges and holes. For example, the below picture shows I haven't done it yet on the right of the selected vertices. This is why there is a pinch here and we will have to create a row of faces around this hole. This will be the air outtake for brake cooling.



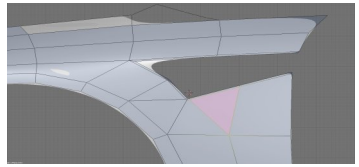
It's time to deal with the air outtake. First, I roughly create the lower faces after deleting one useless edge on the wheel arch. It's not really detailed at this time because of the lack of vertices in the newly created faces. Fewer vertices is a good thing, as it helps controlling the mesh flow, but too few vertices won't allow you to control anything.



Before sharpening the corners, we need to add the row of faces along the air outtake. To do that, I add extra rows of vertices around it and weld vertices when needed to make sure these rows have the same count in the border. In this case, I deleted the horizontal edge going from the upper corner of the outtake and added two rows of vertices: one above and one below the outtake. I only had to merge the two front vertices to one, resulting in a clean flow.



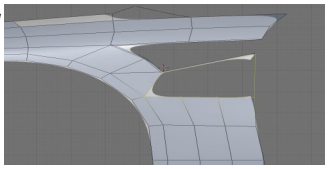
Now it's to sharpen the corners. To do so, I add more vertices near the corner ones. The sharpness of the corner is determined by the distance of these new vertices, closer vertices means sharper corners. Next we have to get rid of the triangles introduced by this move because triangles are always poorly handled by the surface subdivision.



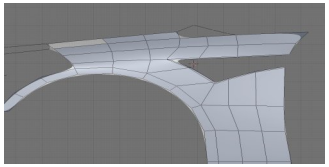
For example, removing the triangle shown above in pink is easy: I just have to add more vertices below the outtake so that there is a one-to-one relationship between vertices (i.e. no triangles, only squares). I removed the vertical row of vertices (in fact I merged it with the neighbors) so I reverted to the previous mesh. Then I added two rows of evenly distributed vertices using Ctrl+R and

3D WORKSHOP : Modeling a High-end Car

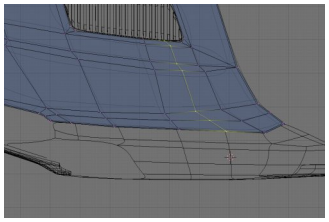
mousewheel. Now we only have to close the faces with the selected vertices shown in the picture.



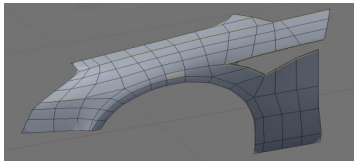
To sharpen the other corner, I added 2 extra rows of vertices, as shown in the picture. When you have to do so, remember to keep a well-balanced mesh by moving other vertices as needed. But also remember that when you move vertices, you have to check the smoothness of your mesh flow again. Yes it takes time, but it's required if you want to achieve a clean job.



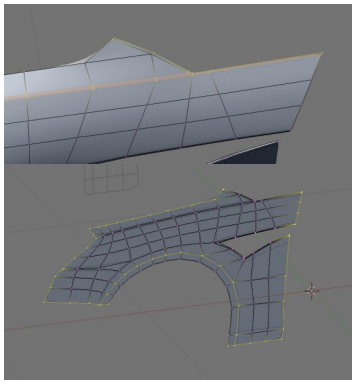
One of the two extra rows I added is shown by the cursor in the next picture. You will notice that it results in adding a vertice on the edge shared with the hood. So the curvatures of the 2 edges (the one on the fender and the one on the hood) won't be the same unless you add the corresponding vertice in the hood edge. To do so, the easiest way is to add a full row of vertices in the hood, as shown in the picture.



I've speed up a little bit on mesh completion, as it's not really different from the previous part. This will allow us to spend more time on a few different issues.

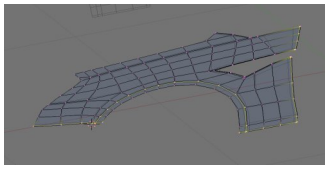


The above picture shows the completed mesh, but it's far from being enough to call the fender finished. The first step is sharpening all the edges. The above picture shows the already hard edges in yellow. Please notice how the upper edge gets wider from middle to front in order to smooth the hard edges. The same goes for the below edge where it gets wider from the middle to the rear. On the top, please note the polyflow. It's easy to choose to achieve either a hard or smooth corner. Here's a detail for reference use.

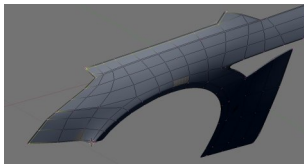


The next picture shows all the still-to-be-sharpened edges.

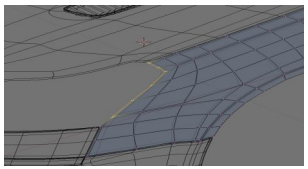
Some are quite easy, they are shown below and selected in yellow. Those edges just need to be given an extra row of vertices along them. Once again, I used loopcuts (Ctrl+R) in non-proportional mode to be able to place the new row 5mm away from the original one. By the way, I tried to use the 2.50 Series of Blender and as of writing the latest release (2.54beta) doesn't allow the non-proportional mode yet. So it's not an urban myth, if you're like me and a heavy non-proportional loopcut user, you'd better stick to 2.49b for modeling.



The next picture shows the more tricky edges. For these ones, please remember that we need to create a gap between the fender and neighboring parts (hood, bumper). So for this reason, I created two extra rows of vertices. The first one is created 8mm from the actual border, and the second one 3mm from the actual border. Now we'll just have to delete the original border so that we have a new border which is 3mm from the neighboring parts and which is followed by a row of vertices 5mm away.

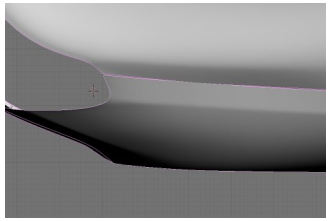


However, this move leads to a mistake behind the not-yet-modeled headlight. We pushed the border backward by 3mm, so we now have a shift between the front edges of the hood and fender. The next picture shows the vertices that need to be moved forward by 3mm.

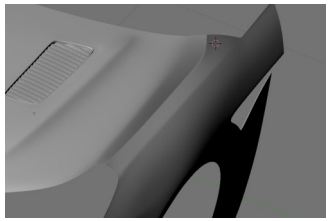


The below picture shows how the shift is now gone. It also shows how unclean the gap is

between the hood and fender. This is because when modeling the fender, I had to add extra rows of vertices to get a clean and well-balanced mesh. Just like we encountered earlier, this led to adding vertices on the border of the fender which is shared with the hood. Now that the two borders are different, they don't align anymore once the subsurf effects are visible.

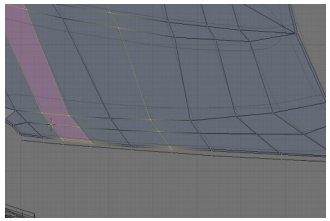


And we have another issue with the hood-fender duo. There is a hard edge on the hood, another one on the fender and a hard area between these two edges where it should be smooth and clean (see the following picture).

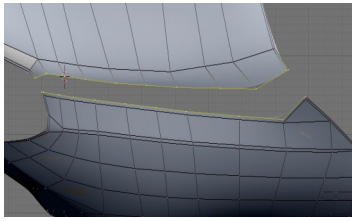


It's no big issue actually. It's just a side effect of modeling one surface at a time instead of modeling the whole car in a single piece. The way to solve it is pretty intuitive. We'll merge both meshes to create a single surface, smooth it and then separate it back to 2 parts.

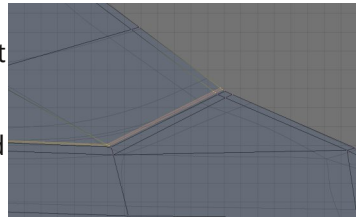
Because of issue #1, the two parts can't be merged cleanly because the number of vertices are different. So just like we did in a previous step, we'll add more rows on the hood which will give us the same amount of vertices on both borders.



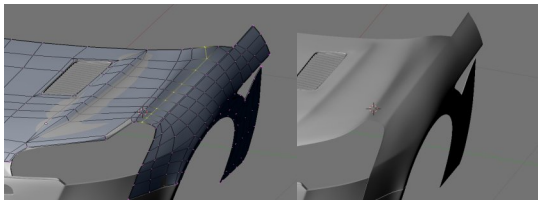
Once done, I just delete the border from the hood (and the row of sister vertices 5mm away). This gives us a wide gap which will be filled with faces. Select both edges, and press the F key. The bottom option in the popup menu will automatically create the faces between the two rows of vertices.



Once done, I make some loopcuts in the middle of the gap between hood and fenders. The first loopcut is 3mm away from what was previously the fender border followed by a loopcut in the middle.

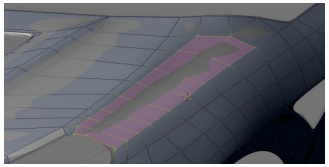


Keep that middle row and remove the neighboring rows: one from the hood, two from the fender. Now the mesh flow is clean, but only to better show how the surfaces aren't clean.

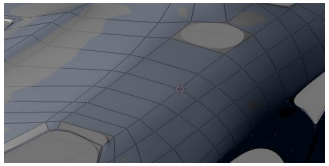


Faster chapter: the rest of the main body

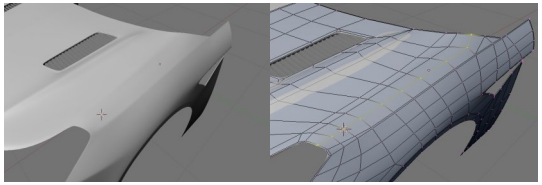
I'll go faster on body modeling, as it's always the same moves. First, remember that we had to clean the area near the hood and fender. The faces we need to work on are the following.



When having that kind of issue, I find it more efficient to remove the middle vertices, then make sure the neighboring curves are clean and then recreate the deleted vertices with a loopcut. It will give us a smoothed row of vertices that we'll move as a single item to adjust the curvature of the surface.

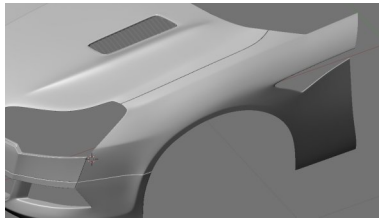


Once the vertices are recreated and the surface smoothed, we get the following result.

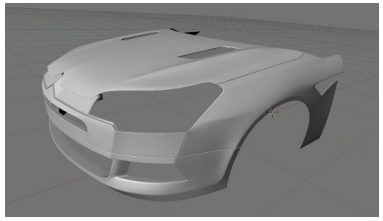


3D WORKSHOP : Modeling a High-end Car

It's now time to split the hood and the fender back to two separate pieces. I won't detail it with pictures as it's always the same move. Do a loopcut at a given distance to specify the width of the gap, then separate them by removing the gap faces, add extra rows on each part to prepare a nice edge for when we will add thickness.



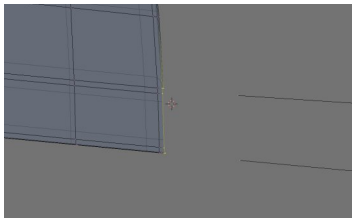
Now with thickness added, the fender can be called finished. At least for now, it's definitely possible to notice bumps later, so we could possibly have to get back to work on the fender – or any other part.



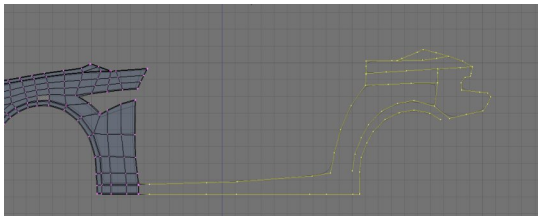
Now it's time to deal with side skirts. When starting a new part of the body, remember it's always the same workflow:

1. Duplicate borders from already modeled neighboring parts
2. Duplicate every other curve from the cage (including the borders from non-yet-modeled parts)
3. Gather all the borders and curves in a single object, and merge the vertices if needed
4. Start building your mesh

About the side skirt, I've been rushing a bit on the fender. The bottom rear side need to be extruded to form the side skirt. So I deleted the thickness vertices from fender on that area. The following pictures show the vertices from the fender which will be extruded to the side skirt. Please notice that the vertices on the right are from the side skirt cage.

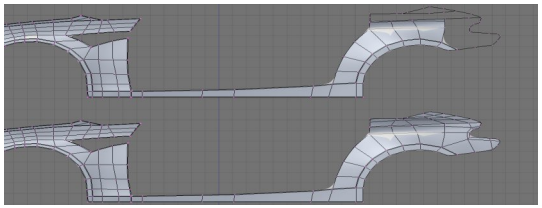


Once connected, the whole part looks like the following picture. In terms of industrial feasibility, it's definitely not realistic. The front and rear fender and the side skirt can't be produced as a single part, it would cost too much to produce or to repair. But as I'm only making a concept car which will never leave the Blender bytes, who cares?

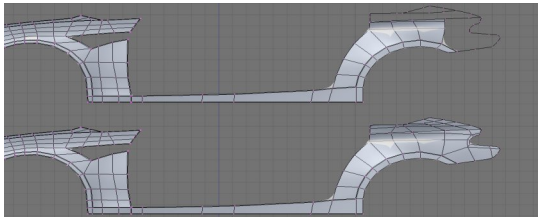


Time to actually create the mesh. As usual I do the easy parts first, where the polyflow is intuitively built from the existing mesh and the cage elements. This is the higher part of the following picture.

The lower part show the fully modeled surface.

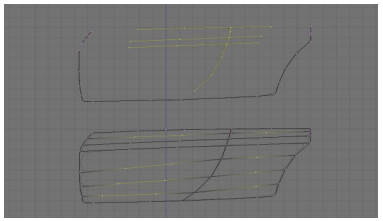


Now it's the usual trick: sharpen borders and edges, and add thickness.

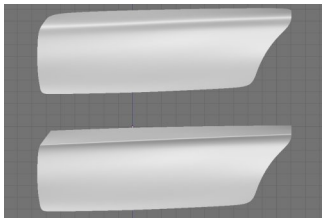


We're done with the fender. Let's take care of the doors. Workflow is the same. The higher part of the following picture shows in yellow the vertices from the cage, while unselected vertices are from existing neighboring parts. The lower part of the picture is more interesting.

The selected vertices are the first I create to ensure the straightness of the doors' shape. Yes, the shapes are more simple on the doors than every other part we already modeled. Building the mesh that way is the best way to make sure the surfaces will be very clean.

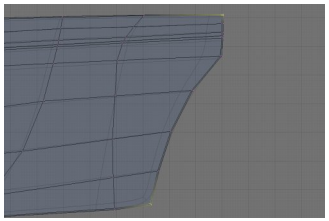


3D WORKSHOP : Modeling a High-end Car

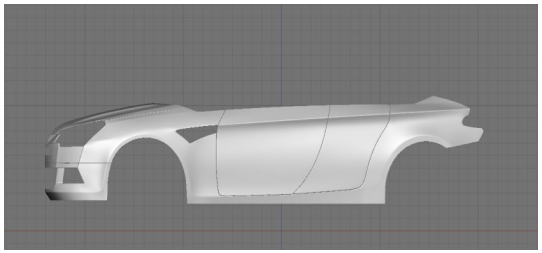


Some pictures show the usual rhythm: sharpen edges, borders. The only difference is to take care of the sharpness of the fold, as it should not be arazo r blade.

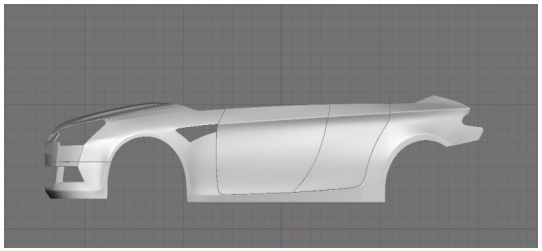
Another thing worth noticing is how the upper corners of the doors are sharp while the lower corners are rounded. The following picture shows the detail of the mesh structure to achieve those 2 different type of corners.

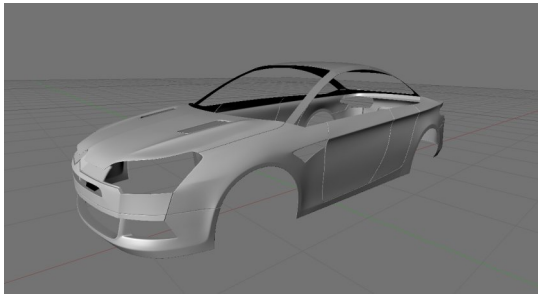


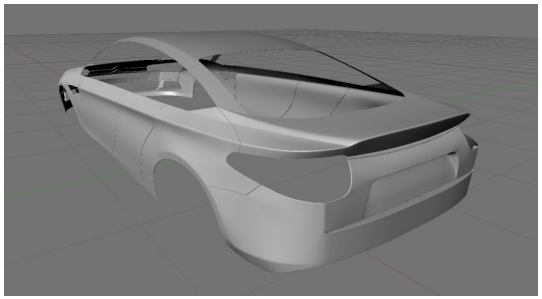
I won't detail separating doors and adding thickness, it's exactly like I explained it before. So we'll move on directly to the finished doors, presented with the rest of the model.



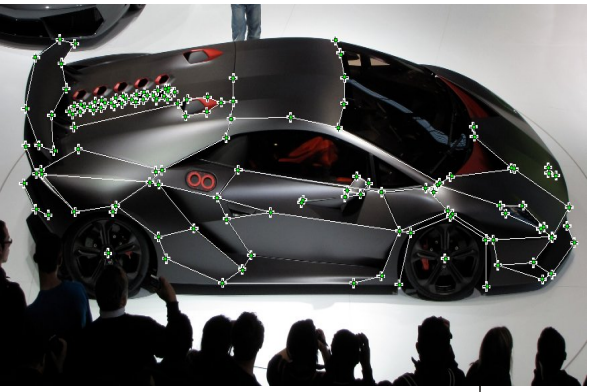
Now you should have reached a significant level of knowledge to know by yourself how to complete the roof, roof arches, windshield frame (really simple parts) and the rear end (trunk is easy, rear bumper a bit more difficult) Here's my very current stage of modeling for this project; the main body is almost done •







3D WORKSHOP : Modeling without Blueprints



by - **Thomas Baron**

Introduction

Modeling an object is the art of recreating it in digital, 3D space. No matter how accurate you want your model, no matter how fast you'll be working, it will always be about these famous 3 dimensions.

Some objects are so simple you can create them from scratch just by studying it and taking measurements. Take a look around and you'll see plenty of these kinds of objects. But there are also many difficult objects with elaborate shapes. Put your ruler back into your schoolbag, it won't be of use this time.

The main solution to modeling a difficult object is blueprints. These aren't blue anymore like architect drawings from 50 years ago but are usually orthogonal (front, rear, side, and top) views of an object. With them you can locate any specific point of the object in 3D space.

But for some objects, there is just no blueprint. So what do we have left? Pictures. It's easy to have pictures of the object you want to model. You can take them yourself with your digital camera, or you can grab tons of them from the Internet. Some people can use them to create a model by eye, but it requires some serious attention due to the nature of perspective. Perspective makes closer things bigger, so the proportions can't be trusted without correcting for perspective. Here comes the approach I wanted to introduce; using dedicated software to make this correction.

How does it work?

Consider our 3D space, where the object stands, and a set of cameras around it. Each camera takes a picture. Put each picture in front of its camera on a transparent sheet. When you look from the location of the camera, the picture will perfectly match the object behind it (if you don't consider the optical distortions induced by camera defects).

Now consider a specific point on your object. E.g. a corner of the windshield, or the center of a rim. You can see this point from some of your cameras, so it shows up on some of your pictures. You can draw a straight line going from a camera to this point and it will pass through the related picture (standing in front of the camera on its transparent sheet) precisely on the spot where this point is located in the picture. It's the notion of projection of the real world on the picture.

insight3d works using this projection, by reverse computing the projection. Give it the pictures, a set of specific points on your model and it will be able to compute the location of the different cameras that shot those pictures.

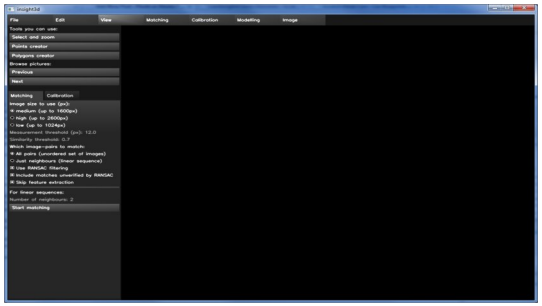
Once the camera is located, it can compute the 3D location of the specific points, by throwing lines from the camera location to the location of each point on each picture. For a given specific point, all the lines will converge onto the real place of this reference point on the model. As a result you're able to compute a 3D model from a set of pictures. Of course it's a simplified 3D model. e.g. I only used 171 specific points. But it would be basic to draw a better spline cage.

Using insight3d

insight3d is Free Software, available for free at http://insight3d.sourceforge.net for both Windows and Linux. Once installed and executed, it shows up with its unusual interface.

Well start straight away. I assume you have already gathered the reference pictures of your object. Mine is a

3D WORKSHOP : Modeling without Blueprints



concept car only shown at a couple of motor shows so there is no blueprint available. I'll add my first picture, using the menu item **"File > Add Image"**. Tip: you can zoom the image using the scrollwheel and pan it using the middle button.

Use the "Points Creator" button on the left allows you to place the reference points on our first picture which are represented by little crosses.

You can see on the picture that a little popup will appear with the zoomed-in portion of the image under the mouse cursor to help you place the points more precisely. It will conveniently auto-disappear if you zoom in

enough. Nice attention from the developer.

Add another picture now. It will appear but the reference points don't show up as you have to place them. insight3d must also be able to do the connection with the ones you placed in first picture.

To do so, use the PageUp and PageDown keys to go through the list of existing points. They will appear on the left of the mouse cursor, conveniently allowing you to see all the occurrences of a given point in all the pictures you have already added. The picture below shows two things on the popup: the zoom and the sample of the previous picture showing this point.

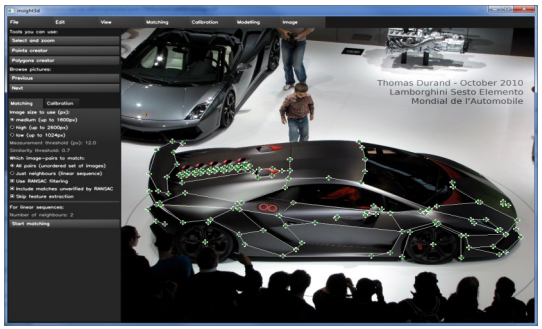
When moving the mouse over a cross, the popup shows again to display the same point in all the previous pictures. For example, the previous picture shows two thumbnails in the popup: the location in the previous picture of the reference point we're about to place in this picture and the zoomed portion of this picture under the mouse cursor. When you have placed this point, the popup window will auto jump to the next reference point. Nice attention from the developer again.

It's now time to place more points to get enough data so that insight3d can compute the camera locations and triangulate the points. You can try that whenever you want by using the menu item "**Calibration >Automatic calibration**", and then the item "**Modelling > Triangulate user vertices**".

Once these two steps are performed, insight3d should display green dots near the crosses, to indicate where it computed the position of our reference points from the reconstructed model.

You may have noticed that the reference point on the right (near the popup) has no green

dot. This means that the current amount of data (number of pictures and number of points) wasn't enough for insight3d to compute the position of this reference point in 3D space. This is easy to fix by simply adding more points on more pictures. As an example, my final set is using 171 points on 15 pictures for the real project about this concept car.

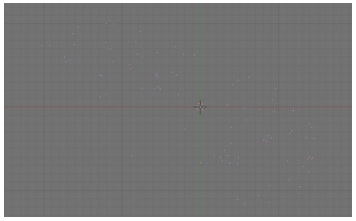


The green dots may also be slightly off the cross. This means insight3d discovered an incoherent location of the reference point on this picture. The green dot shows the place where the reference point should be according to insight3d.

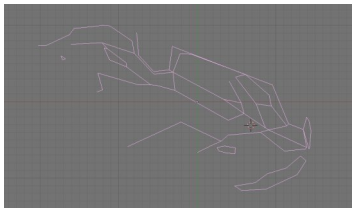
It may be right or wrong, because the computation performed depends on how precisely you placed points on all the pictures. Don't hesitate to zoom in to locate points really carefully as it will help insight3d to reconstruct a more accurate model. Another good thing is to use many pictures. Theoretically, 2 pictures are enough to perfectly locate a point in 3D space because there will be only one intersection of the 2 lines. But as placing a point on a picture perfectly is not possible, many pictures leads to a beam of lines to compute an average position which is much more reliable because the errors are minimized.

Exporting the work to Blender

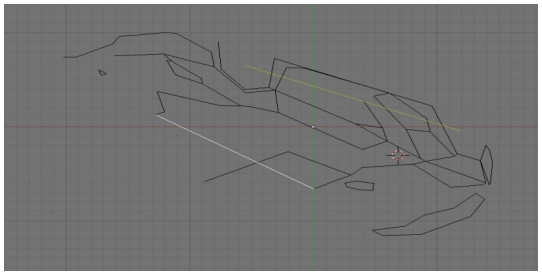
First, you export the model from insight3d. I used the VRML format, by using the menu item "**File > Export VRML**". Then, in Blender, use the menu "**File > Import > X3D & VRML 97**" (the last entry in Blender 2.49b) Do not use the "**File" > Import > VRML 1.0**", it won't work. When done, a cloud of vertices shows up in Blender.



Unfortunately insight3d didn't export the edges, so you'll have to recreate them. Enter Edit mode (Tab key) and create edges by selecting two vertices and pressing the F key. When done, you'll have the following (this picture shows a work-in-progress version of my project, so there is less than 171 points).

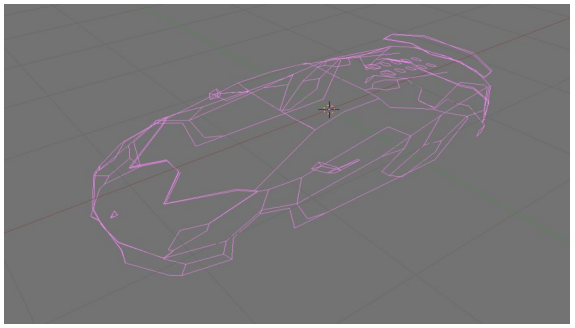


Because insight3d doesn't know about the ground and the natural orientation of our model, it appears randomly oriented. So we have to align it properly. For that purpose, I used the yellow edge in the following picture.

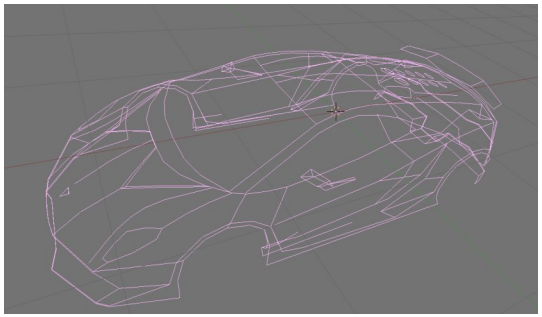


This edge goes from the center of a wheel toth e other one. So it defines horizontal and rear-end orientation. I use this property to align the model. Also, pay attention to the middle vertices (shown in the white line below), they should be aligned as much as possible.

Here's the full model (171 dots) once properly orientated, untouched from insight3d except for a mirror modifier. You can clearly see the proportions look right and the shapes are accurate. It's up to you to use more points to extract more details from the reference pictures and to bring them to your 3D model. For example, here's the current look of my project (as of writing). I just used some more



points, and added details into Blender.



Some extra features within insight3d

Automatic reference points.

insight3d features the ability to discover reference points automatically based upon picture pattern recognition. This works better with pictures that are similar and have few changes between them. The tutorial available on the insight3d website demonstrates this feature for buildings, and there it works well. For my example, it didn't work as I was using various points of views and several backgrounds behind the model (different motor shows). So I had to create all my reference points by hand. Make sure to take a look at the tutorial on the insight3d website if you want to try it.

Polygons.

insight3d allows you to create polygons to be exported to Blender. I didn't use this feature that much, except to create lines (2 vertices polygons) to follow the body lines of the car I was working on. The key tool for that purpose is the "Polygon creator" button on the left side. When activated, you can select a number of reference points, and a polygon is shown using the points you selected. You can confirm this polygon and move on to creating the next one by pressing the Enter key. At any time, you can cycle backwards through all the created polygons using the BackSpace key. Use the menu item "Edit > Erase current polygon" to delete the polygon shown in pink.

Cameras.

insight3d also supports camera export, but unfortunately I didn't achieve a useful result. The exported data for cameras (location, rotation) doesn't match the imported model. insight3d does support exporting to other file formats, for example the .rzi files used by Image Modeler, a similar program by AutoDesk. Opening the .rzi file in ImageModeler (export by insight3d) didn't work either. I guess there are different coordinate systems for the different exports available from insight3d, but I didn't try to read the source code to understand the relationships between these coordinate systems. Save often.

There are some minor bugs with the current version of insight3d (0.3.2). Even though it's a small version number, it's already quite stable. However don't forget to frequently save your project.

One final thought

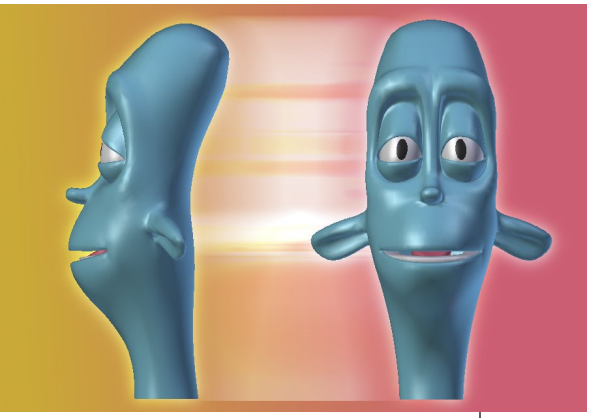
One critic about insight3d, about feedback & support. During my project, I encountered several bugs. An annoying one is that the project file doesn't support the use of spaces in a folder names (at least in the Windows version I used). I tried to reach the developer about that, but I got no reply from him. It looks like he doesn't reply to incoming emails about his software and a comment on a BlenderNation article about insight3d confirmed my fear. Let's hope this article can cause him to change his mind!

http://sourceforge.net/projects/insight3dng/

Picture credits

The first picture of a Lamborghini Sesto Elemento (shown twice in this article) was found in Wikipedia, the author is Alainrx8. The other two pictures of the Lamborghini Sesto Elemento were shot by Thomas Durand, known in the modeling scene as AMV12. Congratulations to both of them for these nice pictures •

3D WORKSHOP : Dynamic Facial Control Interfaces



by - **Pedro Bastos**
Xenxo Alvarez
Veronica Orvalho

Introduction

Character facial animation is difficult because the face of a character assumes many complex expressions. To achieve convincing visual results for animation, digital artists need to prepare their characters with sophisticated control structures. One of the most important and efficient techniques to achieve good facial animation is the use of facial control interfaces, also called facial user interfaces, or facial UIs.

Common facial UIs

But what exactly is a facial UI? Facial UIs are GUIs (graphical user interfaces) with panels and controls to facilitate the animation of a character's face. Facial UIs are usually placed side-by-side with the face of the character in order for the animator to have easy access to them. The panels in the facial UI move in 2D space, as well as the controllers placed inside each panel. The spatial transformation of a control in a given panel will affect the transformation of a corresponding part in the skeleton of the character's mesh. What happens is the controls in the facial UI manipulate the character rig structure which in turn deforms the mesh.

The number of panels in the facial UI usually corresponds to the number of facial regions of the character that need to be animated. Usually the controls are grouped based on the anatomical areas of the face. There are a number of references to help the character rigger do this better, such as Paul Ekman's book, "Facial Action Coding System." Figure 1 shows an example of common panels placed in a facial UI.

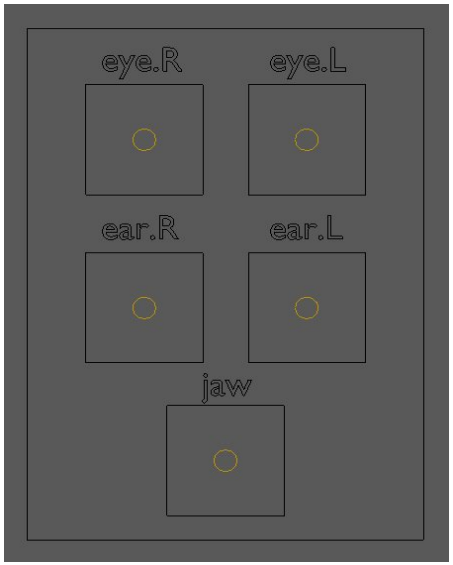


Figure 1 – A common facial UI

Each of the five panels in the facial UI shown in Figure 1 has a controller the animator can freely manipulate within the boundaries of the corresponding panel. The default location of each controller is in the center of the corresponding panel.

Idea of dynamic facial UIs

The facial UI method presented in this article is dynamic because it extends the usability of the common facial UIs used in character facial animation. It is a method developed with Blender's potential allowing an animator to easily relocate and re-dimension a facial UI. The techniques explained in this article allow the animator to customize the limits and the layout of the facial UI and therefore have more control over the animation of the several regions of the face of a character.

Please notice that this article doesn't go into precise detail on how to do common tasks in the Blender software. It is expected that the reader is experienced in Blender 2.5 but if further help is needed just send me an email to ptbbastos@gmail.com. So let's start! First off, get a character's face model. Figure 2 shows the front and side views of the character used in this article, a funny cartoon character called Blue.

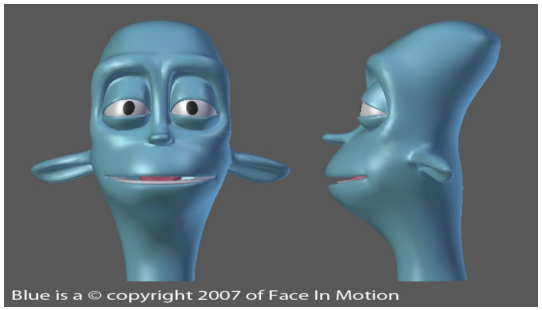


Figure 2 – The character, "Blue"

We will build a dynamic facial UI example for Blue's jaw. Once you understand the method used for the jaw, you can extend it to all other facial regions and build a complete, dynamic facial UI.

Begin by adding an armature to the scene and place two bones to gain control over the jaw. Have a look at Figure 3 to get an idea of how to place the bones. Bone A is the jaw bone itself and bone B is a controller for bone A. Now assign a Stretch To constraint to bone A using bone B as the target. This will allow you to control the jaw with a lot of stretching and squashing abilities.

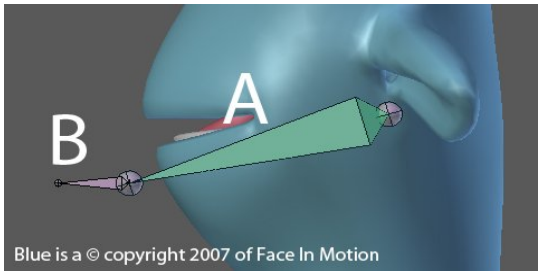


Figure 3 – Jaw bones for Blue

After doing the necessary setup to skin the mesh to the armature (you only have to skin bone A), the time comes to build the facial UI. We are going to build more bones and place them directly in the facial UI. But first we need to build a facial UI with panels so we know exactly where to place the new bones to allow the relocation and re-dimension of Blue's facial UI. Figure 4 shows the facial UI for Blue's jaw, built with two mesh objects and one text object.

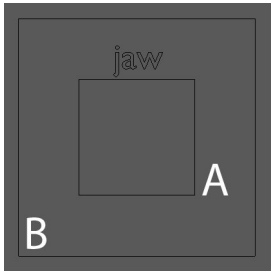


Figure 4 – The objects of the facial UI to control Blue's jaw

We only need a few objects for the facial UI. Objects A and B in Figure 4 are planes configured so the user can only see their wireframe from the front view, the view we are going to be using to manipulate the facial UI. The other object is a text placed above panel A to easily identify that panel, which is the one we will use to restrict the jaw controller. The outside panel, or B, is a global panel to which the movement of panel A will be restricted to, in order to better keep track of the facial UI. The objective is for the user to be able to relocate and re-dimension both these panels. Although in this article we only illustrate the process for the jaw panel, you can later extend it to the global panel as well. So go ahead and place the facial UI side by side to the face of the character. It doesn't need to be very close, nor too far. See the final UI example for Blue in Figure 5.

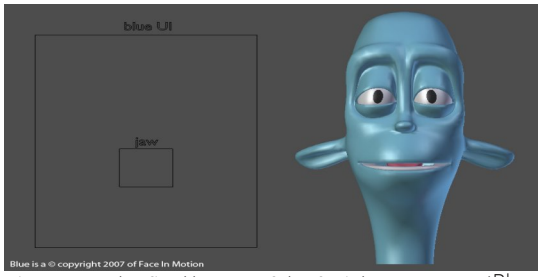


Figure 5 – The final layout of the facial UI to control Blue's jaw

The objects that make up the facial UI have been slightly changed since Figure 4. The outside panel (global panel) is now vertically bigger to better suit the proportions of Blue's face and I've added one more text to better identify the global panel. The next step is to add six more bones to the armature and place them as shown in Figure 6. You can use the 3D Cursor to facilitate this process.

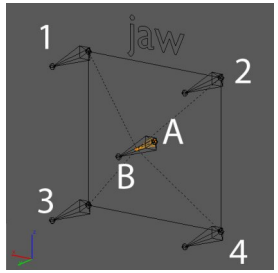


Figure 6 – A six bone setup to control the jaw panel

Bone A, the highlighted smallest one in the center, is the one meant to drag the entire jaw panel. Bone B is the controller of the jaw's animation. The movement of this bone will be restricted to the jaw panel. The other four bones are placed in the corners of the panel and you guessed it, each will control one corner of the panel. Make sure that bone A is the parent of bones 1 to 4 and also of B

(keeping the offset between them as illustrated by the relationship lines in Fig 6).

The next step is to restrict the movement of bones 1 to 4 and also of bone A to the large panel, the one with the text "blue UI", seen in Figure 5. Remember that the main purpose of this panel is to hold all the panels of the different facial regions in order to keep track of the entire facial UI. So, no panel should go outside this one. We do this using two bone constraints in Pose Mode for each of the bones previously mentioned. Be aware of the difference between object constraints and bone constraints in Blender.

The first constraint is a Limit Location, to prevent the bones from moving in the Y axis (you can see the spatial orientation for Blue's facial UI by looking at the mini axis in Figure 6 but you have to be aware that the orientation in your scene may be different from this one). The second constraint is a Shrinkwrap targeting to the global panel object. Remember to repeat this process for bones 1 to 4 and also A, so each of them has the two types of constraints mentioned. Figure 7 shows these constraints types.

Figure 7 – The types of constraints to use

We can't forget to restrict the movement of bone B to the jaw panel. So let's assign another Limit Location constraint and one more Shrink-wrap constraint to bone B. Notice that this time the target of the Shrinkwrap constraint is the jaw panel, not the global panel. At this point, if you move the bones in the system, the panel doesn't follow along. In order for that to happen, we can use a number of techniques, such as Vertex Groups. But I find it to work better using the Hooks modifier.

So go ahead and assign four Hooks to the jaw panel object. Do the necessary configuration of the Hooks modifiers keeping in mind that each Hook should point to the correspondent bone. Figure 8 shows the Hook modifier's options. Send me an email if you have trouble setting this modifier.

Figure 8 – The Hook modifier

Now you can drag the panel in Pose Mode using the facial UI bone system (using bone A in Figure 6). But there's one object being left behind: the jaw text. For the text object to follow along, you can simply make it a child of bone A. But if you want you can also make it dynamic using another bone to control it. Notice how now you can also change the position of the corners of the panel using the corner bones in Pose Mode. Next we have to link the jaw bone in the facial UI to the jaw bone in Blue's face mesh, so we get some motion of the jaw. Figure 9 shows the bones we need to link.

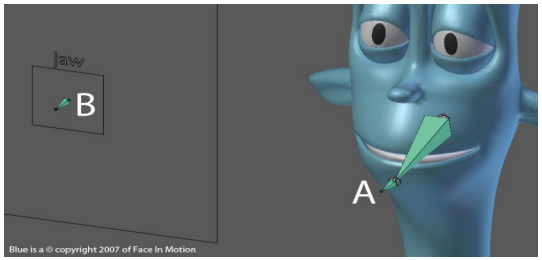


Figure 9 – In this case bone A is simply going to mimic bone B

The purpose is for the jaw controller bone in the mesh (bone A in Figure 9) to mimic the movements of the control bone in the facial UI (bone B in Figure 9). We can do this by assigning a Copy Location constraint to bone A and using bone B as the target.

To wrap-up the process, you can help the animator by hiding the unnecessary bones and assigning custom shapes to the bones the animator is supposed to control. This will improve their appearance and make the facial UI a lot more user-friendly.

Conclusion

This method adds a lot of flexibility to common facial UIs, especially the UIs using lots of panels for every facial region. And even more functionality can be added to it. Figure 10 shows a final configuration result where the jaw panel was first moved up and then its

bottom border was extended down to allow the jaw controller to be dragged further down and create the expression of surprise on Blue. Now imagine using this method with a series of panels to control the eyes, eyebrows, ears and other facial regions of this character.

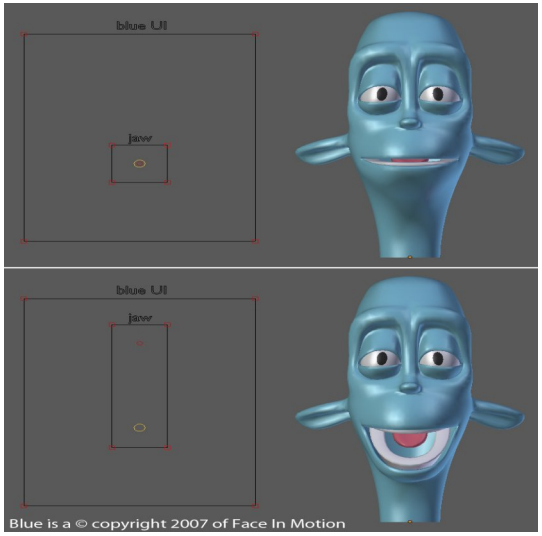


Figure 10 – Using the dynamic facial UI built for Blue'sf ace

I hope you liked it. If you had difficulties setting up the dynamic facial UI, you can email me with questions and I will be glad to help on the more technical details •

Acknowledgments

"Use the provided video file "dynamic_facial_ui" to see the method working in Blender. The character Blue is a © copyright 2007 of Face In Motion."

MAKING OF : Alpha Polaris - Making adventure game graphics with Blender

by - **Teemu Vilén**

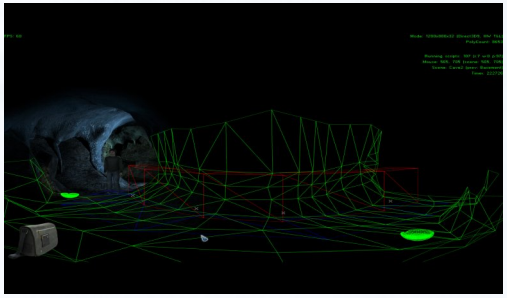
Introduction

Alpha Polaris is a point-and-click horror adventure game developed by Turmoil Games and set to be released in June 2011. We have been working on the project for roughly two years now, using Blender for visuals and Wintermute as a game engine. The core of the game is the titular Alpha Polaris, a distant oil research station in the icy wastes of Greenland. Our team of six people set out to produce a consistent, high-quality gaming experience with a special focus on the place itself. Our setting wasn't going to be a "slog through different levels" world, but an isolated setting full of little details. In this article, I'll discuss some of the visual challenges we have tackled.

Making adventure games is a very art heavy business. To pull one off with 1280x800 wide-screen graphics, pre-rendered backgrounds and real-time characters, we needed a lot of art resources. This meant we all had to learn how to do modeling and texturing in Blender. In the beginning, only our art director Lassi had significant modeling experience. I

think this is one of the things we have been very good at. Throughout the project, Lassi has been very effective in training people to use Blender. With a limited initial skill set and a team with no game industry background, we've also approached the project very iteratively, improving the graphics and re-doing things when needed.

Our key art assets in Blender are the scenes. With indoor locations, one Blender scene corresponds to one game screen. Outdoors, there might be several screens rendered from one Blender scene. For example, our station from the outside is a full 3D environment. One of the more important observations we made regarding the pre-rendered scenes is that including dynamic elements will make them more interesting. We don't use parallax scrolling, so we employ different sprite animations like snowing overlay, glow from aurora borealis, subtle flickering on computer screens and so on. Illusion of depth is also vital. Since the backgrounds are really just 2D pictures, they are projected onto hidden low poly 3D geometry. The resulting character masking and shadows add to the illusion very effectively. With hidden 3D geometry masking the pre-rendered background, a realtime character can walk around an object as if it was really in 3D space. Typical for adventure games, we also rendered separate front masks for scenes.



One would initially think that working with such a well defined setting is relatively simple when it comes to rendering. However, our scenes contain a high number of interactive

MAKING OF : Alpha Polaris - Making adventure game graphics with Blender

and changing details, and are revisited in different parts of the game. On top of that, we have three totally different lighting conditions: morning, evening and night. Combined with relatively long rendering times, it has been an arduous process. For example, if an artist forgot to render a night version for objects on the kitchen table, the error would be painfully obvious.

All rendering was done with Blender's own renderer. We went through several other possibilities, but at the time none of them were integrated well enough into Blender to use in a heavy project without problems. For example we needed to render lots of sprite animations for the scenes with render layers and alpha. The scene renders were all post processed in Photoshop. Luckily it has good batch tools so post processing the sprite animations was also possible.

The same scenes were also used for cut scene videos. The challenge was to build the suspense while adding visually dynamic content to the game. Instead of elaborate storyboards, the team prepared test animatics for the cut scenes phase of the project which meant we already had a lot of the scenes done. This sped up the process considerably. The cut scenes in the game are not very long, but combined they total at over 12 minutes, so they were a big part of the project for a small team.

With characters, the most significant problem was getting them to Wintermute. It uses the age-old DirectX format for models, and the Blender .X-exporter didn't seem to work correctly. As usual, this meant some semi-blind pipeline testing and frustration. If we got to the point we could see the characters in Wintermute, they always had some lighting and animation related glitches in them. After about two weeks of testing, we ended up using Gandaldf's excellent .B3D exporter, and then converting them to .X in FragMotion.

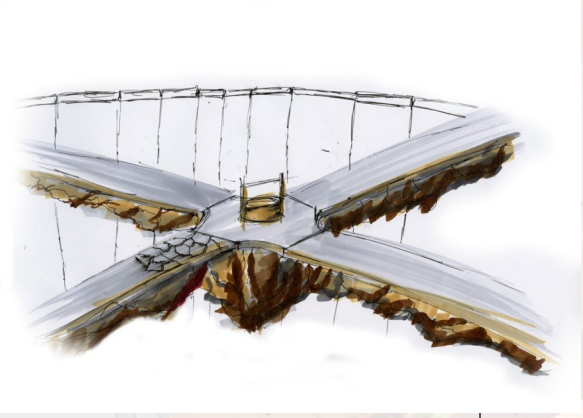
The characters in the game are realtime with 3000-4000 triangles. Wintermute only supports diffuse maps, so we had to bake and paint some lighting effects on the character textures. They are 2048x2048 resolution and were mostly painted in Photoshop, but Blender's texture paint was also used as it is excellent for clearing texture seams with the clone stamp brush.

Character animations were pretty straightforward to do. We made one action strip per animation in Blender and combined the animations in Fragmotion. Our characters are semi-realistic so the animations had to be close to a real person's movement. Animating the walk cycles was the hardest part. We videoed our team walking on a treadmill for reference at a local gym which helped a bit.

I'm happy to say that there's not much to do anymore. At the moment I'm writing this, we are implementing the localization kits - the game is to be published in several languages by a major publisher. Our long, iterative process seems to be coming to conclusion. We will definitely continue to work with Blender in the future, as it's proven to have outstanding features and flexibility for indie game development, not to mention the community support •

Writer: Teemu Vilén, Project manager of Alpha Polaris and CEO of Turmoil Games LTD

MAKING OF : The next small step with Blender3d



by - **Alexander Weide**

Introduction

Hi, I am Alexander from Leipzig and I had the idea to build a game in that sunny place of Germany. But how? First I have to say I was working in the film industry for one year along with a couple of years in commercials and other 3D animation stuff using tools like Houdini or Maya. But one day I thought: it's not what I want. We live in a world with perfect animations and "perfect" pictures, but is that what we want? We want to have entertainment, we want to do anything, we want to build our own worlds. For only one person it is an impossible task to build effects driven movies like District 9 or Battle L.A in a short period of time, without thinking of budgets. So it was my decision to build one world - alone. I will see how far I can get, how much I can reach, and how fast I can reach my target.

I am an artist rather than a programmer so Blender was the best choice to help me meet my goals. The new version has a very high potential to build games with all that graphic stuff we know from World of Warcraft or other MMOs. So it's possible to build an RPG game alone with all that stuff too. All Blender users know the game Yo Frankie!, and it was the first big step in my opinion.

Thanks to the hours they spent to build such a good game like Yo Frankie! with Blender, it gave me the energy to start my own project. As I started it was not easy to find a good story, so I began with drawing. After a couple of days in January 2011 I got the direction, a short story and personally it's what I want to spend time on.

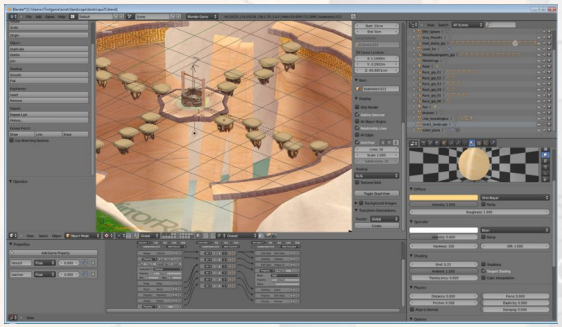
The creative part of it is very similar to creating 3D animations. I played a lot of games, I saw a lot of movies and I was looking at what the community does. And the community does a lot, but most don't reach the end of their projects. Why? Is it not possible to do it? Is Blender not the right tool? I don't know, but I will complete the end of my experiment and I hope it will inspire other artists around the world to do same with Blender!

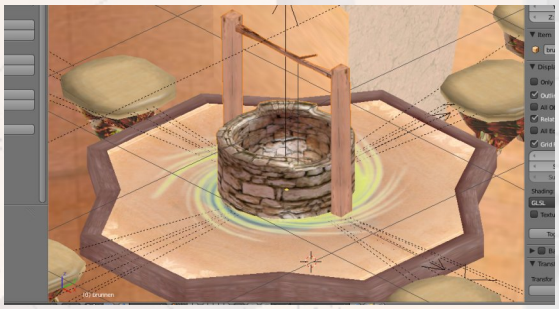
In the next few days I will start a Facebook site to share the project with the community. If the community stands behind me, I will gather more power to go forward. If the community is not interested in that, it could be a hard way for me maybe. So I hope I will get a lot of feedback. For now I will not share any blender files, scripts, textures assets, so don't ask for it. All files are done by myself so I have a lot of things to do in the next couple of months.

If my concept works in a browser with the webplugin Burster for Blender I will present it as Browser game in 3D. I got a lot of ideas. I hope the community brings the feedback and support I need for that.

Deadline is around Fall of 2011.

Information about it is on Facebook.





MAKING OF : The Crown

by - **Stephan Rodriguez**

Introduction

Hello, my name is Stephan Rodriguez and I'm a a French 3D artist. I live in a very attractive and sunny town in the south of France called Montpellier with my wife and 2 children.

Since I began 3D back in 2001 when I discovered Blender, my goal has always been to do some animation one day. At that time I was amazed by every 3D short film I could see but I never imagined I could do the same a few years later. I began like every other Blenderhead, reading every tutorial I could find on the web, asking questions on forums like Elysiun (the ancestor of Blenderartists) or a French forum called zoo-blender where I used to spend a lot of time.

I'd say it took me about 4 years to feel really comfortable with Blender, just enough to make some decent looking still pictures. I had learned to model, texture, light and was beginning to produce some very basic animation along with some particle and camera work.

In 2005, after learning some rigging/skinning stuff, I began to work on shapes (called rvk's at that time) for facial animation of my characters. Today this is still the area that provides me the most pleasure. I love the way it gives life to characters and this is where the acting job begins.

So that year I produced my first short animation with a lipsync job to celebrate the

10th Blender anniversary. This was called "Happy Birthday Mr Blender" and is visible on my Youtube Channel. Tony Mullen, the author of "Introducing Character Animation With Blender" even gave me the pleasure of including it on his DVD and I was very proud of this.

During all these years, I had a couple of story ideas running in my head. Then in 2009, after improving my animation skills, I decided to step forward and lay down my story on paper. On February 2nd of 2009, my short animation project officially started.

"The Crown" tells the story of a young girl who has been revered as a Mini Miss in her town. What she doesn't know is that the crown on her head, found 50 years ago in the countryside, was actually lost by some distant visitors and they are here again today to take it back ...

The short will be (hopefully) humorous, which is why I decided to model some toony and quite ugly characters, especially the young girl who is supposed to be pretty (she's a Beauty Miss).

One of the first things to do besides modeling was the storyboard. This is a very important thing because you constantly refer to it to know what scene to do, what the shot will require in terms of animation, characters, set, props etc. I created a personal

MAKING OF : The Crown

storyboard model that fit my needs which describes the action for each scene and shot, the camera moves and the effects (focal blur, particles, wind etc). The storyboard also lets you know exactly what you have to model, what will be visible on screen and helps avoids useless modeling.

Another tip when you undertake such a project, is to publish your work in progress on a website forum. This forces you to work as viewers ask you about your project when you haven't been posting for a while! It also helps keep your motivation intact when you hear good things about your work. A good and constructive criticism helps you to improve your skills and be aware of small defects on your pictures. You wouldn't normally see these because you're constantly watching your own work.

Starting this project has required a huge amount of dedicated hard work for the last 2 years, but it was a good decision considering the pleasure it brought me as well as the opportunity to improve my skills in several areas. Character animation is something I really want to pursue in the future.

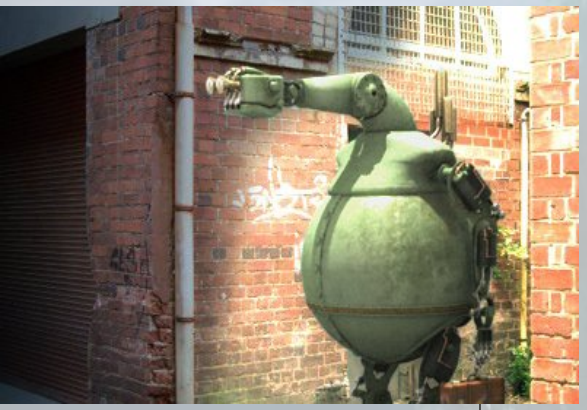
I've made about 2min 50sec of film so far, with a target of 5 seconds of rendered footage by week, but I sometimes reach 7 or 8 seconds a week. If everything goes well I hope to finish the short in June of 2011. The sound design will come next and hopefully I'll be competing for the Suzanne Awards in October.

Hope you'll enjoy watching my short film.

Regards,

Stephan

MAKING OF : Gordy



by - **Darcy Brooker**

Introduction

Gordy – A couple of years ago I started sketching out some ideas for an illustrated story book about a Steam-punk styled robot called Gordy. Working as much as I could in my spare time I estimated that the project would take up to two years or more to complete. It seemed a little overwhelming but I decided to persevere anyway.

I had soon developed a basic outline of the story and jotted down a few good creative visual ideas. The next few months were spent modeling and experimenting with new techniques for the look and feel of it and creating some preliminary art. Then one fateful day I discovered the Flash game Samorost, a simple point and click adventure game using layers of animated cutout props and characters. The wheels in my head were sent spinning. Immediately I thought to myself, I could do that! Why make a book when I could make an animated book!?!

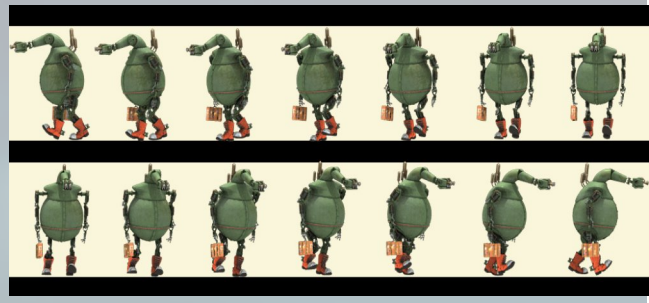
This was going to be an even bigger challenge than the book but I reasoned: I can use Flash, I have game industry experience, I have written music and can do the art so why not?

The pre-production planning went into overdrive. This time I had Gantt charts, asset lists, puzzle designs, a design document and a more thoroughly revised story. As soon as enough of the first level planning had been completed, I was away! Full steam-punk ahead!

Six months later I had created almost four detailed scenes, rigged and animated the main character, rendered dozens of character animations, and got the whole lot working in Flash. It was a mammoth personal achievement, but there was still a nagging doubt in the back of my mind, and when I looked more seriously at what I had accomplished compared to what still needed to be done, there came the realization that I had bitten off way more than I could chew. It would take at least half a dozen artists another three years of full time work to finish this project.

I have since revised my goals and scaled things back to a much more manageable level, and have decided to go back to the idea of creating a book. In the six months or more I spent working on the game I have created artwork and assets that can be used in the book, learned a truck load, and am very much looking forward to this next creative adventure.

The style of the artwork in the book will be a combination of Steam-punk, Raygun Gothic, Indian and Riven-esk. Whether or not the book has any text is still something I'm not too sure



MAKING OF : Gordy

about. I really just want to make a book full of eye candy about a person's experiences while on holiday. A simple narrative that could be conveyed in a series of pictures. I'm sure it will all work out :)

The book will be released independently through Createspace and will also be available through Amazon. I will be announcing its completion through all the popular Blender websites as soon as that magic day arrives and I will be posting artwork and updates on the book at my DeviantArt page at...

http://darkladder.deviantart.com/

and at my personal webpage at...
http://www.darcybrooker.com/

MAKING OF : Navigating Carnegie Mellon

by - **Brian DeVincentis**

Introduction

Two other college students and I are developing applications to help the Carnegie Mellon University (CMU) community navigate the campus. Behind the scenes, Blender is allowing us to efficiently gather the necessary data.

Overview

The goal of the Navigating Carnegie Mellon (NCM) project is to create free, open source mobile and desktop applications to show those unfamiliar with the intricacies of the campus how to go from one place to another in the shortest amount of time. It displays graphical and textual directions to users overlaid on Google Maps and blueprint images. Our desktop application will actually be a web application so that it can be quickly and easily accessed by anyone. The mobile application will have the same functionality with the added ability for the user to follow along as they walk the route. The application is currently under development. This summer it will be released to the Android marketplace and deployed to www.ncmproject.org.

Map Generation

A significant amount of work went into generating the map. A complete map needed to be generated to tell the application how to get around campus and how long it takes to do so. This map allows the application to take every possible route into consideration when deciding which route is best. The map is simply a series of points and edges (points being locations and edges being connections between points). I had access to all the blueprints for each floor for every building on campus. For each floor's blueprint I created the points and connected them together. Then I connected all of the floors together.

Next, I associated information such as room numbers and terrain with each point and edge. The room numbers allowed the application to associate the user's input with a point. The terrain data indicates whether an edge represents stairs, an elevator, flat ground, etc.

Also, we needed to collect the time required to walk across all the connections on the map. This allows our application to determine the fastest route between two places and also give the user an approximation of how long it will take to walk the calculated route. The map, room associations, terrain associations, and timing data are included in a set of files to be imported into our applications.

Image: Points and lines overlaid on a blueprint.

Why Blender?

So, why did I use Blender for the map generation? Well, it was actually a critical tool that managed and supplied tools for generating the huge map covering the entire campus.

To create the map of points and connections I simply used the background image feature to draw points and edges over the image using the mesh tools.

The interesting part is what happens next – utilizing Blender's Python Scripting API. The API's ability to access any part of Blender from the UI to the underlying data was the reason we chose Blender. Another great

MAKING OF : Navigating Carnegie Mellon

aspect of the API is that it uses Python which is fast and easy to write. All this made it possible to create a custom set of tools quickly and easily that were made specifically for the NCM project. Without Blender we probably would have needed to write a program from scratch which would have significantly increased development time.

The first script I wrote added user interface elements, speeding up the generation of the map and the process connecting room names to the points in the map.

Another script I wrote exported the data out of Blender to our own custom file format. This was very useful because it eliminated any complications in moving data from Blender into our application. In fact, it allowed us to make a custom file format that made file sizes as small as possible and also include extra data such as room names and timing data.

To collect the timing data we actually created an Android application which made this process very fast (a large building with 10 floors can be timed in 1 hour). The Android application generated files in a different format which were then imported into Blender. I wrote another script that imported these files and integrated the timing data into the existing blend files. Then the times could be exported along with the map via my file exporter.

Image: A screenshot of the Android application displaying directions.

Want to Help

If you have any questions, comments, or suggestions we would love to hear from you. Go to www.ncmproject.org (will be created by summer 2011) for more information and to find out how you can help. If you want to contact me, feel free to send me an email (bdevince@andrew.cmu.edu).

The Future

1. Feature Additions – a variety of features that would make the applications more complete.

2. Other Platforms – bring the Android user experience to other mobile platforms such as the iPhone or Windows Phone. These could replace the mobile web application.

3. Other Campuses – bring the application to other campuses. We would find volunteers to map their campus and submit the data to our team to be included into the application.

4. 3D – make a 3D version of the desktop application to give users a better view of the campus.

Going 3D?

Currently we are developing 2D applications for Android and the web (both a desktop and mobile application). Originally we were planning to create a 3D web application. However, as we did more research we found that it simply is not feasible (yet). Three dimensional applications on the web currently have poor performance, but new technologies are under development that may make this application possible. A new web standard called WebGL is enabling developers to write 3D apps for the web that have better performance because this technology takes advantage of hardware acceleration. Flash is also working on a version of the Flash Player plugin that utilizes hardware acceleration which will enable 3D capabilities.

Blender surely is not the component holding us back. We are eager to take advantage of some of its 3D capabilities. In fact, we have already figured out how to make a simple 3D model of the entire campus. We are obtaining the CAD files for the blueprints of every building. We import them into Blender via the DXF importer. Then using a script I wrote, the blueprints are simplified and the lines are extruded up to make walls. For texturing, multiple textures are put onto the model of the floor and then baked to increase performance. For the outside portions of campus I obtained a 3D terrain model of the campus from Google Earth. All of this can be exported to almost any format because Blender supports many file types.

Once 3D web technologies are widely available we will still be challenged with generating a more detailed 3D model of the campus. It is not feasible to generate a detailed 3D model of the entire campus manually. We welcome suggestions regarding methods/technologies that would allow for the automatic creation of a detailed, aesthetically pleasing model of the campus •

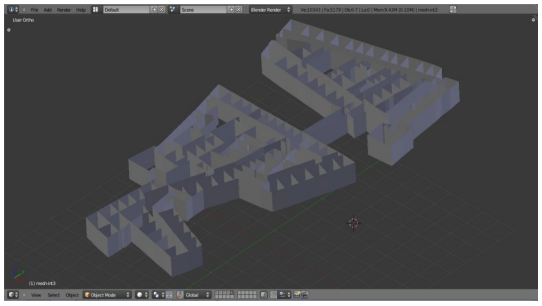


Image: A 3D model of a floor derived from a 2D CAD file.

MAKING OF : Pahelika Revelations : Insider's Peek

by - **Rohit Bhonsle**

Introduction

Pahelika Revelations is a follow up of our successful title Pahelika : Secret Legends. Its a casual Puzzle Adventure game, set to be released later this year.

This game's story continues from where the previous game had left off. The theme of the game wraps up the player in an adventure to save the human way of life from an ancient evil accidentally unleashed during his adventures in the previous title. Our game designer has blended puzzles and varied locations in an intensely gripping storyline. To improve player immersion, special care was taken to make an intuitive interface for the game. On the technical side, the game uses a heavily modified version of the free PopCap Framework.

Art Pipeline

The art pipeline itself is quite straightforward. The game designer describes a location and makes a rough sketch. Since we do not have a dedicated concept artist at IronCode (we're a small company), the rough sketch is taken by our lead artist who fleshes out the details and then the location is modeled and rendered. All locations, scene and inventory items have been modeled, textured, rendered using Blender's Internal rendering engine.

Since the first day Ironcode started, we have used Blender 3D to create the in-game graphics, and in the same tradition of Pahelika Revelation uses lots of 3d rendered artwork in a 2d game with all the graphics and sprites pre-rendered and overlaid with a custom particle system based special effects inside the game.

Instead of using the stable version of Blender, the team used the cutting edge 2.5x version, which turned out to be a good decision. Apart form Blender, both Gimp and Photoshop were used in creating the textures along with Inkscape. We aimed to outdo ourselves with this game, and with the improvements in Blender, we were able to achieve quite a lot.

Blender trouble

Blender2.5x bought some difficult moments to the work, It was quite difficult to deal with alpha-mapped textures alongside Blender's Sun Sky features. The resulting white artifacts gave a very difficult time to the graphics team. In the end we worked around this limitation by compositing the scene by first separating offending items to different layers and after separate render passes later joining them to create the final piece.

Image: Store room.

Sometime a weird bug cropped up in one of those constant upgrade builds we compiled and optimized, which lead to reorientation of camera

Image: Hidden Chamber.

projection and position despite it being locked completely.

Productivity Improvements

Blender2.5x was also a lot more productive than the earlier version. It has a better, more streamlined interface, that allows us to work faster. It is also more flexible than the previous version. The advancement in functionality and usability was felt and welcomed with open arms by our art department.

Image: The Monk.

Blender2.5x was a lot faster in rendering compared to earlier versions and this came as a boon to us. In most cases we were going about 10-20% faster in just about every rendered scene as compared to the previous version. Custom compiling it to our machines led to even better render times.

Image: Guard room.

Conclusion

Whereas Blender proved to be an extremely capable tool when modeling and animating, it was found wanting on the Rendering side. Our journey in creating the artwork for this project is still very delightful and I sincerely hope that everyone will enjoy our game as much as we are enjoying making it ●

BLENDERHEAD : From a Blender Beginner

by - **Avish John**

Introduction

I am an artist. I have been passionate about sketching since childhood and as I grew up I got interested in digital art. I have had no prior formal education in 3D. As 2011 dawned, my resolution was to learn 3D. My next question was which software I should learn. I had earlier tried working on Maya, as that should have been a very obvious choice. But there was a small community at my work place who used a software program called Blender. The only thing I knew about it was that it was open-source. I was apprehensive about whether I should really trust this software. It didn't look very impressive. Still, I wanted to try my hand at it. I spoke to my colleague who is a Blender expert. He told me Blender could do so much. It depends on the user. I took his word and I downloaded a copy of Blender 2.56

It's been three months now and the Blender experience has been splendid. It is not just the software, but the community as a whole that has made me love this software. I was surprised to find a compositor and a video sequencer built inside a 3D package. I searched for tutorials and I found virtually everything I wanted to learn. I was able to pick it up quite quickly. The user interface which I once thought was 'not impressive' turned out to be pretty intuitive. In two months time I made my first complete Blender render – 'Headphones'.

I appreciate the work that the Blender community is doing in developing and providing a tool free of cost for the cause of art. I want to continue to learn Blender and someday make my contribution to this community •

Avish John

avish.john@gmail.com

avishjohn.blogspot.com

BLENDERHEAD : A Big Fan

by - **Alex van Ophuizen**

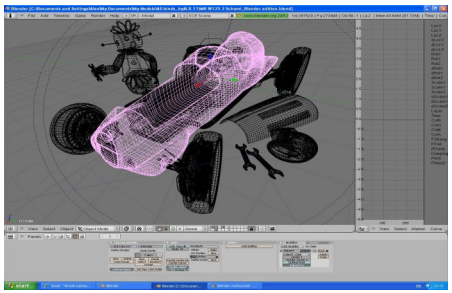
Introduction

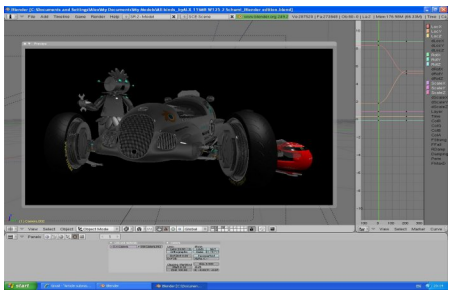
Hi everyone! My name is Alex. I'm a soon to be 33 year old graphics designer from the Netherlands. I never write any articles on software or other tools, but since it's for Blender...I might have a go!

I came in contact with 3D in general somewhere in 2005. A friend of mine showed me "Sketchup", a 3D modeling tool from (back then) @lastsoftware. It's a great, easy to learn program! If you like making architectural stuff, this is a great way to do it! Most people know it from working with Google Earth. It has is limitations though... (so do I, for that matter, haha), and I wasn't able to create nice organic shapes.

The same friend pointed me to 3ds Max, but I didn't like it from the start. I stopped trying soon after. He apparently saw some potential in my skills, and kept pushing me to try something else. He showed me the Blender.org site and even bought me the official Blender 2.3 manual! Ever since then, my poor girlfriend doesn't get as much attention as before! I can really lose myself when "playing" with Blender.

I'm not an expert, but for now I pretty much know how to model anything. Ic an texture it, do some basic rigging (I'm gonna buy the DVD on rigging soon!) and some animation. I know some basics of the particle system and had some fun with the fluid-engine. My computer is not state of the art, so due to lack of computing power, I'm a bit stuck at a certain level.





For now I just love the basic modeling. It gives me enough room to create what I want! I don't use any outside rendering programs, I do everything with Blender internally. I also posted a render for "Everything but the kitchen sink".

It's a remake of the robot from "The Jetsons" known as Rosie. She's working on a W125 Mercedes Benz F1 car from 1937. Rosie was a matter of a few hours, the basic car took me about 3 days. It has suspension, and through some parenting and tracking, it really works! Rendering this picture on 1280px took about 3 minutes, so it's too heavy for doing some animation (for now!)

To sum things up, I can't really point out a favorite tool, except for the handling of Blender in general. The way the mouse works (reverse) and all the keyboard shortcuts... it can take some time to get used to, but it's sure worth the effort!

I will never work with another program again! (Seriously!)

My compliments to every modeler out there who made it to the Blender gallery. It's quite amazing what you guys posted!

Kind regards, a big fan,

Alex van Ophuizen

"aka" DesignsbyALX

KNOW HOW : Eight Things I Have Learned Making a Short Movie

by - **Enrique Sahagun**

Introduction

I recently finished my first long project. It's a short film called 21:00 [1-3] made with Blender 2.48. When I started working on it two years ago, I used PovRay for all my digital works. I knew that this project was going to be too complex to be made with PovRay so I started looking for an open source Maya-like suite. Then I found Blender and it was like an epiphany.

As you can imagine, the last two years have been a long learning experience. On the one hand I discovered the huge technical power of Blender. On the other hand I had to learn how to make a movie and take the story I had in my mind and get it into frames. After these two years I haven't learned enough about Blender to make a good technical tutorial. What I can do instead is give you some advice on how to face the problem of making a movie.

Of course, I will talk about my own experience which is that of a single guy making a nine minute short film, from the script to the final sound edit. I'm not a film director and my short film is made of mistakes over mistakes. My only aim with this advice is to help you save time by avoiding some of those mistakes.

1. Write a good solid story

The first step is to have something to tell. When I started thinking of making this movie, my first approach was to make it with a real camera and real actors. But some scenes were too dangerous to be made with real people so I decided to make it digitally. I changed the technique, but the story survived that change.

What I mean is that the key point of making a movie is to have a story to tell. It's easy to find videos from digital artists that are technically jaw dropping. Many short movies are just incredible visual exercises. That's OK and sends a great message that this

can be done and I'm good enough to do it. But a car transforming into a robot or a house burning is not a story. Besides, if you have a good story, technical deficiencies are less important.

During a long project, something terrible is likely to happen to you. At some point, you will get bored of it. To maintain your own focus during a long time, it's important for you to believe in your own project. So take some time to write a good script.

2. Sketch a Storyboard

Creating a storyboard is something everybody suggests and no one does but this is particularly important for digital films. Sketching the movie with simple drawings is essential to develop the visual appearance of your film. Even if you know exactly how the story is going to develop, it's important to understand that there are still thousands of ways to transmit it through a film. The duration of the shots, the position and motion of the camera relative to the action, the lighting and the use of flashbacks can all dramatically change the perception of the story. A storyboard is essential to state the final visual aspect of your movie. Plus it also helps to keep things tidy (see point 5).

But in a digital film there is an even more important reason to make a storyboard. If you know where the camera points in every scene and how wide the shot is, you'll know from the

very beginning which and how many elements will need to be modeled. And believe me, this will save a lot of time. For example, when I started working on my film my instinct told me to fill the 3D space with lots of detailed objects as if it was a real space. However many of these objects did not appear in the final film. If you first think where the camera is going to point and what the frame contains, you'll be able to make a precise list of the models you'll need.

3. Learn how to move the Camera

One thing I learned making this movie was how important the camera was. It may seem that knowing the story you want to tell and its evolution in time is enough to make a movie. You could place the camera or move it with no rule but to capture the action, and your film would be OK. But this is completely wrong. The first thing you notice when you make your first digital shots is that they don't look like those you see in real movies.

It is important to understand that the camera critically affects the way a story is told. Where the camera points, its position and its motion, all add a new layer of perception to your story. This perception has been defined along one hundred years of movie making. That's why it is important to learn the basics of camera framing and the meaning of shots. There are lots of articles on the Internet on the subject [4-7] and also comments on the technique of particular directors [8]. But I recommend a simple exercise. Take a film you like and watch it carefully paying attention to the position and motion of the camera. You will learn a lot of useful tricks just by studying a single movie.

4. Don't respect the general lighting.

Don't be afraid to change the lights for a single shot. In a shot, lights can be divided into two groups: general lights and correction lights. The first group are those defined by the physics of the scene (the sun, a lamp, the radiosity, etc). The second group are those that belong to a particular shot and they are used to correct the lighting of the first group when needed. If your shot is not properly illuminated, don't change the configuration of the general lights; use the correction lights. Actually, this is the technique used in real movies.

But be careful. You can change the light, but at the same time it is important to respect the shadowing of the general lighting and the general colour palette of the movie. If you make strong changes in the lighting between two shots, you can break the connection between them [9].

5. Keep things tidy.

Think of this. You want to make a 4 minute short film. That's six thousand frames (at 25fps) and about 20 to 40 different shots. My approach is to create a blend file for each shot (previously defined in the storyboard). This allows me to keep things tidy. If you save each shot in a different blend file, you can correct them (if needed) by changing a small number of keyframes. On the other hand, if you create a very long sequence, with camera changes in a single file, a simple modification in the middle of the action can be painstaking.

Another interesting technique is to create a master blend file with all the pieces of your animation as they are rendered in the video sequence editor (VSE) to have a global idea on how things are going.

6. Previews, previews and more previews!

Make as many previews of your shots as you can. Not only for single shots but for whole sequences. Prepare several shots, "make a camera view render" and put them together with the VSE. This will give you a perfect idea of the quality of your shots and will allow you to make corrections on the camera motion or on the duration of the shots before you tackle the final high quality render.

7. Take advantage of post-production

When I started working on 21:00 I didn't pay

much attention to the Node Editor. This was completely wrong. The light and textures you use on your scene are very important. and their final aspect can be improved or powered using the Node Editor. In addition to tuning the brightness, contrast and blur, you can also make sophisticated colour corrections. So before making the final render, play a little with the Node Editor.

The same happens with the VSE. Explore its possibilities because it allows the addition of very complex effects. For instance, the muzzle flash of a machine gun can be painted in the GIMP and then added using the VSE.

The message here is the following: don't lose your mind trying to model or simulate

everything. There is probably an easy way to do it using a post-production trick.

8. The Sound and Music

Sometimes when you watch some finished shots you think: "this is not what I was expecting." For example, if you animate a guy shooting a gun, or a car skidding, the final result sometimes looks a bit absurd. The animation then seems to be wrong. Sometimes it's because it lacks the sound effects. The sound not only empowers the images, it also completes their meaning and helps the audience to understand and believe what they are watching.

The same happens with the music. Music helps to drive the feelings of the audience to where the director wants them to be. A sense of sadness, violence or stress can be created by choosing the right piece of music.

The Film •

http://vimeo.com/20367421

References:

1. www.enriquesahagun.es
2. 21:00 (English subtitles) http://vimeo.com/20367421
3. 21:00 http://vimeo.com/20366008
4. http://www.siggraph.org/education/materials/HyperGraph/animation/cameras/traditional_film_camera_techniqu.htm
5. http://www.desktop-video-guide.com/fundamentals-of-camera-movement.html
6. http://www.aber.ac.uk/media/Documents/short/gramtv.html
7. http://www.mikejones.tv/camera-in-motion/
8. http://borgus.com/hitch/hitch2011.htm
9. http://en.wikipedia.org/wiki/Match_cut

BLENDER BYTES : RenderWeb: Tapping In To New Resources

by - **Nathan Moyer**

Early March of 2011 marked the launch of the new social rendering platform, RenderWeb (apps.facebook.com/renderweb). The viral expansion of the RenderWeb Facebook App immediately began changing the way Blender hobbyists and small-studio animators render their projects. In essence, RenderWeb is a free online rendering farm for the Blender animation software, powered by social relationships within Facebook.

By integrating community-based rendering within Facebook, RenderWeb has simplified the process of both volunteering computers and harnessing potential volunteers. With the simple click of a button, numerous friends, family members, and colleagues can donate their computer's extra cycles to render animations.

RenderWeb not only allows the collaborative effort of rendering projects, but also instantly shares newly rendered animation videos with the community. Once rendered, the original animator can download a ZIP file containing the high quality images while the whole community begins to share and discuss the newly rendered animations. RenderWeb also offers the ability to download the web-friendly M4V video file so that animations can be posted to other locations (like YouTube or Vimeo).

For those projects requiring more privacy, rest assured that there is a private setting for uploading and socially rendering animation projects. While private projects have the lowest priority within the queue, if multiple personal computers are volunteered to render that private project, a "private" render farm can be created.

From a Volunteer stance, RenderWeb allocates projects based on the existing relationships within Facebook. Volunteers render their own projects first, their friend's projects second, then random public projects within the queue and finally, private projects. Thus, the more friends an animator has, the higher the potential for computational power. Volunteers can also directly select a project of their own choosing from the queue to render.

By utilizing the RenderWeb Facebook application, a whole world of untapped computational resources can be allocated. There are no installs, no complicated instructions, and no complex procedures limiting accessibility. Anyone can volunteer, and anyone can render.

Since its launch, RenderWeb has been creating a roadmap for the future. Over the past two months, the community has offered numerous suggestions about how to improve the social rendering experience. With this, RenderWeb has begun its natural evolution; responding to the community in order to better the entire social rendering experience.

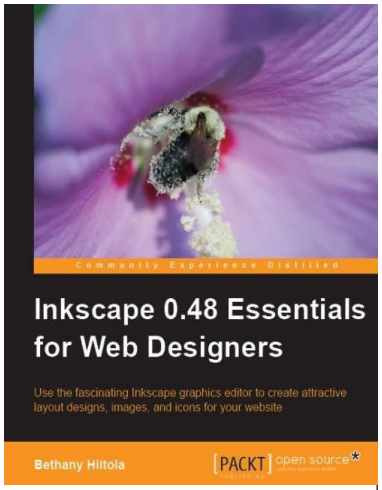
Among the many community-based suggestions, RenderWeb's roadmap includes integration with commercial animation software, additional image output options, a standalone applet, more user-defined controls, and a finely tuned media-centric layout. Further down the road, RenderWeb is also interested in opening up the data to make it more available for external developers. This would allow for plugins, widgets, and possibly a RenderWeb screensaver for passive volunteering.

We hope that the integration of Facebook with social rendering will redefine social computing. RenderWeb connects animators with new friends, new communities, and new computational resources. With this shared, community effort, rendering will no longer impose a bottleneck in our creative and 3D pipeline. We will all have ample computational power to render everything that our production and creativity demand.

The RenderWeb Team is currently comprised of Adam McMahon (developer/founder) and Nathan Moyer (designer). Adam McMahon is a PhD candidate at the University of Miami, and Nathan Moyer is owner and designer of Light Made Liquid, LLC ●

The RenderWeb Team can be reached at Contact@RenderWeb.org.

BLENDED REVIEW : Inkscape 0.48 - Essentials for Web Designers



by - **Gaurav Nawani**

★★★★☆

Inkscape 0.48 - Essentials for Wen Designers.

by: Bethany Hlitola

316 pages

Publisher: Packt Publishing

For a budding artist and or designer nothing comes close to figuring out every nook and corner for features and hidden tools in a piece of software.

The book Inkscape 0.48 Essentials for Web Designers is such a book which such readers will find enjoying simply because it have so much explanation on various features and tools available in Inkscape.

This book starts of by explaining the very basic of Inkscape to users and the way it does will delight the readers who have almost no previous experience to a Vector drawing tool.

Second chapter bring the reader to the main topic of the book designing a website from within Inkscape.

Inkscape offers some very basic set-up for slicing and outputting website code so this does not really conveys much in terms of a tool for a web-designer however the other tools that help him design the various graphic elements required in website are gradually introduced in following chapters.

Most exciting part is the three chapters on 'Text Styling', 'Wallpapers and patterns' and 'Building Icons and Logos'. Since these toics are the one which are most important to an graphic designer, the book here offers a good resource for an beginner to an intermediate level designer. It is structured towards learn and practise and the tasks are not difficult at all at the same time they introduce you to the tips and tricks of using Inkscape features and tools effectively.

Finally it also touches upon Inkscape's geeky side in a chapter that deals with XML Editor in Inkscape. Followed by a small chapter in producing simple animation by combining Gimp into the work-flow and finally a chapter on extending Scribus functionality by Plug-ins and scripts.

A good read mostly for some one looking for innards of Inkscape but not for experienced readers, unless they are turning towards Inkscape after using other vector applications.

One thing that also needs to be said is that this book is not a thorough web designing guide as almost the title seems to imply rather it provides few very basic website examples and how Inkscape's modest slice tool can be of some help, and while doing so it makes you learn Inkscape well.

All in all a good read, so if you think Inkscape is your thing give it a go you wont find it lacking ●

Want to write for BlenderArt Magazine ?

Here is how

Step1. Choose what you want to write.

- Tutorials explaining Blender features, 3dconcepts, techniques or articles based on the focused theme of the issue.

- Reports on useful Blender events throughout the world.

- Cartoons related to blender world.

Step2. Send submissions to sandra@blenderart.org.

- Send us a notification on what you want to write and we can follow up from there.

Step3. Some guidelines you must follow.

- Images should be properly cut and represent the text appropriately.

- Images should be provided seperately in a folder named (images, img or pictures).

- Images should be named/labled likewise (image1 or img1 etc).

- Provide proper captions for images if and when needed.

- Image format prefered is PNG but good quality JPG can also do.

- You can submit inline images in documents like DOC or Openoffice ODT etc but make sure the images were properly names before importing them in docs.

- Images inside a PDF are a strict no, but a pdf document with images if provided to show how the author wants the formating of doc will be appreciated.

- Make sure that screenshots are clear and readable and the renders should be at least 800px, but not

- Text should be in either ODT, DOC, TXT or HTML.

Step4. Archive them using 7zip or RAR or less preferably zip.

Step5. Additional stuff that you can do

- Please include the following in your email:

- Name: This can be your full name or blenderartist avtar.

- Photograph: As PNG and maximum width of 256Px. (Only if submitting the article for the first time)

- About yourself: Max 25 words .

- Website: (optional)

Note: All the approved submissions can be placed in the final issue or subsequent issue if deemed fit. All submissions will be cropped/modified if necessary. For more details see the blenderart website.

BA takes no responsibility fo the material in any form and the submission will automatically mean that you have agreed to the blenderart terms and conditions for submission for more information please do read the disclaimer.

Gordys Home -by Darcy Brooker

Coma Postal Frente -by Martin Eschoyez

Audi in Cathederal -by Nathan Felix

Leopard -by Johannes Ewers

YouCanTakeItWith -by Nick Keeline

Headphones -by Avish John

Hofner -by Diego De Cata

La Casa Del Lago -by Lauty Tapia

Music-Peppermill -by Christian Bremer

River -by Paulo Bardes

Roise Mech -by Alex

Whiskey Glass - by Adrian Kubasa